EDGE BASED ALIGNMENT ALGORITHM

Donghui Wu

Yushan Huang

FIELD OF INVENTION

[0001] This invention relates to methods for aligning images.

DESCRIPTION OF RELATED ART

[0002] When a photographer captures a scene using a camera, the desired field of view may be larger than the normal field of view of the camera. Digital photography allows a panoramic image to be produced without the need of purchasing special equipment such as a panoramic camera or fisheye lenses. For example, a photographer with a digital camera may capture a series of digital pictures of a scene by rotating the camera and taking pictures in a sequence. The captured images may then be stitched together to produce a panoramic picture of the scene. Similarly, film-based photographs can be digitized, and the panoramic picture can be composed by stitching together the digitized images. Presently, digital image programs are available for stitching multiple digital images together to form a panoramic picture. Exemplary programs include Ulead Cool 360®, Live Picture PhotoVista®, and MGI PhotoSuite III®.

[0003] Thus, what is needed is a method for aligning images.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] Fig. 1 is a camera motion between two images in one embodiment of the invention.

[0005] Figs. 2A, 2B, and 2C illustrate possible alignments of two images in one embodiment of the invention.

[0006] Figs. 3A, 3B, and 3C illustrate possible alignments of two images in one embodiment of the invention.

[0007] Fig. 4 illustrates a method for determining image correlation in one embodiment of the invention.

[0008] Fig. 5 illustrates two exemplary images to be aligned in one embodiment of the invention.

[0009] Fig. 6 illustrates a correlation map for made by the method of Fig. 4 in one embodiment of the invention.

[0010] Fig. 7 illustrates a method for calculating a correlation value between two images in one embodiment of the invention.

[0011] Fig. 8 illustrates a method for calculating matched edge count in one embodiment of the invention.

SUMMARY

[0012] In one embodiment of the invention, a method for aligning images includes (1) selecting a pair of overlapping pixels when two second image are aligned with a camera motion; (2) if both pixels are edge pixels, incrementing a correlation value between the two images for the camera motion; (3) if only one of pixels is an edge pixel, decrementing the correlation value between the two images for the camera motion; (4) repeating steps (1) to (3) for other pairs of overlapping pixels in the two images to determine the correlation value for the camera motion. The method further includes (5) repeating steps (1) to (4) for other camera motions; (6) selecting camera motions having large correlation values; (7) determining matched edge counts for members of the group; (8) selecting the camera motion with the largest matched edge count; and (8) aligning the two images with the selected camera motion.

DETAILED DESCRIPTION

[0013] Fig. 1 illustrates an example of a camera motion. A panoramic program typically includes an alignment algorithm that estimates the camera motion by searching for the best correlation between two images. Assuming the camera motion between images 12 and 14 is translation and rotation in the image plane, the motion can be described by a three value motion vector: [shift_x, shift_y, r_angle], where (shift_x, shift_y) is the translation vector and r_angle is the angle of rotation. In Fig. 1, (X, Y) are the axes of image 12, (X', Y') are the axes of image 14, and the original of each image is defined at the center of the image. Pixel coordinates between the two images can be transformed using Equation 1.0 listed below.

$$\begin{aligned} x' &= x\cos(r\_angle) + y\sin(r\_angle) + shift\_x \\ y' &= -x\sin(r\_angle) + y\cos(r\_angle) + shift\_y \end{aligned}, \qquad \text{Eqn. 1.1}$$

wherein (x, y) are the coordinates of a point in image 12, (x', y') are the coordinates of a corresponding point in image 14.

[0014] Assuming that a range of the camera motion is known, the alignment algorithm can search in this range for the best correlation between the two images to recover the camera motion. One conventional way to define image correlation is as follows:

$$Corr = \frac{1}{N} \sum_{px\in W} |I(p) - I'(p')|, \qquad \text{Eqn. 1.2}$$

wherein Corr is the correlation value, W is the overlapping region between a first image and a second image, p is a pixel in the first image located in the overlapping region, p' is the overlapping pixel of p in the second image, I() is the pixel value, and N is the number of pixels in the overlapping region W. Equation 1.2 can be rewritten for a color image as follows:

$$Corr = \frac{1}{N} \sum_{px\in W} \left(|R(p) - R'(p')| + |G(p) - G'(p')| + |B(p) - B'(p')|\right), \qquad \text{Eqn. 1.3}$$

wherein R(), G(), and B() are the RGB pixel values. Equations 1.2 and 1.3 may not able to correctly resolve image correlation in all practical situations.

[0015] Listed below are some practical situations that any image correlation definition must resolve correctly in actual use.

[0016] Moving object in the scene

[0017] In some situations, there are moving objects such as people and vehicles in the scene. These moving objects make the overlapping region between the images dissimilar.

[0018] Luminance change in the images

[0019] In some situations, the luminance will change between the images. Although some cameras provide a panoramic mode, which ensures that the exposure and the focus remain fixed while the camera is in this mode, some slight changes in the luminance still exist in the images.

[0020] Multiple solutions in alignment

[0021] In some situations, there are ambiguities that result in more than one possible solution for image correlation. The ambiguity is mainly caused by the fact that the images form only part of the whole scene.

[0022] Fig. 2A illustrates one example of matching ambiguity where an image 22 and an image 24 both include a block 26. Figs. 2B and 2C offer two possible solutions for correlating the images using equation 1.2. In the first solution shown in Fig. 2B, block 26 is matched in both images. In the second solution shown in Fig. 2C, an overlapping region 28 is matched in both images. Although two solutions are possible, experience shows that most users will choose the first solution because it matches the object rather than the background. This rule is hereafter referred to as "matching object not background."

[0023] Fig. 3A illustrates is another example of matching ambiguity where an image 32 and an image 36 both include an edge 36. Figs. 3B and 3C offers two possible solutions for correlating the images using equation 1.2. In the first solution shown in Fig. 3B, the entire edge 36 overlaps in both images. In the second solution shown in Fig. 3C, only part of edge 36 overlaps in both images. Experience shows that most users will chose the first solution because it matches the most edge points. This rule is hereafter referred to as "matching more objects if possible."

[0024] Fig. 4 is a flowchart of a method 50 for a panoramic program that provides a more robust definition of image correlation in one embodiment of the invention.

[0025] In step 52, the program selects a translation vector [shift_x, shift_y] and a rotation angle r_angle that make up a camera motion. The translation vector [shift_x, shift_y] is selected within a range of [minx, maxx] by [miny, maxy], and the rotation angle is selected within a range of [minr, maxr]. Together, [minx, maxx], [miny, maxy], and [minr, maxr] define a range of camera motions that will be searched for the camera motion that produces the best correlation between the two images. In one example shown in Fig. 5, the program selects a camera motion 62 creates an overlaps 63 between an image 64 and 66.

[0026] In step 54, the program computes a correlation value between a first image and a second image when they are aligned with the camera motion in this iteration. A method for calculating the correlation value is described later in reference to Fig. 7.

[0027] In step 56, the program determines if it has calculated correlation values for the range of camera motion. If so, step 56 is followed by step 58. Otherwise step 56 is followed by step 52 and method 50 repeats until the correlation values for the range of camera motions have been determined.

[0028] In step 57, the program constructs a 2-D correlation map 66 (Fig. 6). The axes of map 68 are the values of the translation vectors of the camera motions. The points in map 68 are the correlation values of the corresponding translation vectors. For all the camera motions having the same translation vector but different rotation angle, the program only maps the motion vector that has the largest correlation value. In other words,

$$Corr\_Map[x,y] = \underset{r\in[\min r, \max r]}{MAX}(Corr(x,y,r)), \qquad \text{Eqn. 2}$$

wherein Corr_Map is the mapped correlation value.

[0029] In step 58, the program selects "peaks" from map 68, and then selects "high peaks" from the peaks. A peak in map 68 is a camera motion that has a correlation value greater or equal to its neighboring points in map 68, and greater than at least one of its neighboring points in map 68.

[0030] In step 60, the program determines a matched edge count (MEC) for each of the high peaks and selects the camera motion with the greatest MEC. A method for calculating the MEC is described later in reference to Fig. 8.

[0031] Fig. 7 is a flowchart of a method 70 for the panoramic program to calculate the correlation value by matching edges between two images aligned with a selected camera motion in one embodiment of the invention.

[0032] In step 72, the program initializes a correlation value Corr to zero.

[0033] In step 74, the program selects a pair of overlapping pixels from in the overlapping region of the two images if they were aligned with the camera motion selected in step 52 (Fig. 4). In the example shown in Fig. 5, pixels p and p' are overlapping pixels in overlapping region 63 between images 64 and 66.

[0034] In step 76, the program determines if both of the overlapping pixels are edge pixels. If so, step 76 is followed by step 78. Otherwise, step 76 is followed by step 80. The program can determine if the overlapping pixels are edge pixels by applying a first order differential edge detection filter such as a Sobel filter. In one embodiment, the Sobel filter is defined as:

$$g_x(x,y) = I(x+1,y-1) + 2I(x+1,y) + I(x+1,y+1) - I(x-1,y-1) - 2I(x-1,y) - I(x-1,y+1);$$
$$g_y(x,y) = I(x-1,y+1) + 2I(x,y+1) + I(x+1,y+1) - I(x-1,y-1) - 2I(x,y-1) - I(x+1,y-1);$$
$$G(x,y) = |g_x(x,y)| + |g_y(x,y)|;$$

Eqn. 3

wherein g_x is the Sobel gradient in the x-direction, g_y is the Sobel gradient in the y-direction, and G is the Sobel gradient in the x-direction and the y-direction. If gradient G(p) is greater than a threshold T, then pixel p is declared an edge pixel. If gradient G(p'), which is also written as G'(p') to distinguish it from G(p), is greater than a threshold T', then pixel p' is declared an edge pixel.

[0035] In step 78, the program increments the numerator of the correlation value because the overlapping pixels are likely to match when they are both edge pixels. In one embodiment, the correlation value is incremented by an edge orientation matching value derived from a second order differential edge detection filter such as a Laplace filter. In one embodiment, the edge matching value is defined as:

$$\frac{2L(p)L'(p')}{L^2(p) + L'^2(p')}, \qquad \text{Eqn. 4}$$

wherein L(p) is a Laplace gradient applied of pixel p, and L'(p') is a Laplace gradient of pixel p', which can also be written as L(p'). In one embodiment, the Laplace filter is defined as follows:

$$L(x,y) = I(x,y) - \frac{1}{9}\sum_{i=-1}^{1}\sum_{j=-1}^{1} I(x+i, y+j). \qquad \text{Eqn. 5}$$

[0036] The value of equation 4 is the largest when L(p) is equal to L'(p'), which occurs when the overlapping pixels are both edge pixels having the same edge orientation that defines what is inside and outside of the edge. Thus, the value of equation 4 is the largest when pixels p and p' are most likely to match. Step 78 is followed by step 84.

[0037] In step 80, the program determines if at least one of the overlapping pixels is an edge pixel. If so, then step 80 is followed by step 82. Otherwise step 80 is followed by step 84.

[0038] In step 82, the program decrements the numerator of the correlation value and increments the denominator of the correlation value because the overlapping pixels are not likely to match when only one of them is an edge pixel. In one embodiment, the program decrements numerator by ½ and increments the denominator by 1.

[0039] In step 84, the program determines if it has processed all the overlapping pixels in the overlapping region. If so, then step 84 is followed by step 86. Otherwise step 84 is followed by step 74 and the program cycles through method 70 until all the overlapping pixels have been processed.

[0040] In step 86, the program stores the correlation value for the selected camera motion.

[0041] In summary, the calculations of steps 76 to 82 (Fig. 7) can be summarized in the equations below:

$$Corr = \frac{\sum_{p \in W} f(L(p), L'(p'), G(p), G'(p'))}{\sum_{p \in W} h(G(p), G'(p'))}$$

$$f(L(p), L'(p'), G(p), G'(p') = \begin{cases} \dfrac{2L(p)L'(p')}{L^2(p) + L'^2(p')} & \text{,if } G(p) > T \text{ AND } G'(p') > T' \\ -\dfrac{1}{2} & \text{,if } G(p) > T \text{ OR } G'(p') > T' \\ 0 & \text{,else} \end{cases} \qquad \text{Eqns. 6}$$

$$h(G(p), G'(p') = \begin{cases} 1 & \text{,if } G(p) > T \text{ OR } G'(p') > T' \\ 0, & \text{else} \end{cases}$$

[0042] Fig. 8 is a flowchart of a method 90 for the panoramic program to calculate the MEC between two images aligned with a selected camera motion in one embodiment of the invention.

[0043] In step 92, the program initializes the value of MEC to zero.

[0044] In step 94, the program selects a pair of overlapping pixels from in the overlapping region of the two images if they were aligned with the one of the camera motions selected in step 58 (Fig. 4).

[0045] In step 96, the program determines if both of the overlapping pixels are edge pixels and have an edge orientation matching value greater than a threshold. If so, step 96 is followed by step 98. Otherwise step 96 is followed by step 100.

[0046] In step 98, the program increments the MEC. In one embodiment, the program increments the MEC by 1. Step 98 is followed by step 100.

[0047] In step 100, the program determines if it has processed all the overlapping pixels in the overlapping region. If so, then step 100 is followed by step 102. Otherwise step 100 is followed by step 94 and the program cycles through method 90 until all the overlapping pixels have been processed.

[0048] In step 102, the program stores the MEC value for the selected camera motion.

[0049] In summary, the calculations of steps 96 and 98 (Fig. 7) can be summarized in the equations below:

$$MEC=\sum_{p\in W}k(L(p),L'(p'),G(p),G'(p'))$$
$$k(L(p),L'(p'),G(p),G'(p'))=\begin{cases}1 & \text{,if } G(p)>T \text{ AND } G'(p')>T' \text{ AND } \dfrac{2L(p)L'(p')}{L^2(p)+L'^2(p')}\geq\dfrac{1}{3} \\ 0 & ,else\end{cases}. \qquad \text{Eqns. 7}$$

[0050] Various other adaptations and combinations of features of the embodiments disclosed are within the scope of the invention. Numerous embodiments are encompassed by the following claims.